UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/726,902	12/03/2003	Mitchell Alsup	5500-88700	4177

53806 7590 02/28/2006

MEYERTONS, HOOD, KIVLIN, KOWERT & GOETZEL (AMD)
P.O. BOX 398
AUSTIN, TX 78767-0398

EXAMINER	
FENNEMA, ROBERT E	
ART UNIT	**PAPER NUMBER**
2183	

DATE MAILED: 02/28/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	10/726,902	ALSUP ET AL.	
	Examiner	Art Unit	
	Robert E. Fennema	2183	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *12/03/2003*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-35* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-35* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)

2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.

4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .

5) ☐ Notice of Informal Patent Application (PTO-152)

6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-35 are pending.

Claim Objections

2. In Claim 27, the phrase "duplicates a trace in trace cache" is believed to properly read "duplicates a trace in a trace cache", and has been interpreted as such for the remainder of the office action. Appropriate correction is required.

3. In Claim 32, Line 5, the phrase: "searching trace cache" is believed to properly read "searching a trace cache", and has been interpreted as such for the remainder of the office action. Appropriate correction is required.

Claim Rejections - 35 USC § 112

4. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

5. Claim 28 is rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

6. Claim 28 recites the limitation "continuing construction of an incomplete trace already in progress" in line 1. There is insufficient antecedent basis for this limitation in the claim, as there is no mention of continuing construction of an incomplete trace in the

specification. Given this lack of description in the specification, it has been assumed throughout the rest of this office action that this was intended to mean that a trace is further built upon, for example, adding a second, third (and so on), instruction to the trace being constructed. Appropriate correction is required.

Claim Rejections - 35 USC § 102

7. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

8. Claims 1-2,11-15, 24-26, 32-34 are rejected under 35 U.S.C. 102(b) as being anticipated by Rotenberg et al. (herein Rotenberg).

9. As per Claim 1, Rotenberg teaches: A microprocessor (Abstract), comprising:

an instruction cache configured to store instructions (Section 2.1, first paragraph);

a branch prediction unit (Section 2.1, first paragraph);

a trace cache configured to store a plurality of traces of instructions (Section 2.2, second paragraph); and

a prefetch unit coupled to the instruction cache, the branch prediction unit, and the trace cache (Figure 4, and Section 2.2. The Instruction latch appears to fill the role of a prefetch unit);

wherein the prefetch unit is configured to fetch instructions from the instruction cache until the branch prediction unit outputs a predicted target address (Section 2.2, where it is said that "on a trace cache miss, fetching proceeds normally from the instruction cache); and

wherein if the prefetch unit identifies a match for the predicted target address in the trace cache, the prefetch unit is configured to fetch one or more of the plurality of traces from the trace cache (Section 2.2, where it is said that "on a trace cache hit, an entire trace of instructions is fed into the instruction latch, bypassing the instruction cache).

10. As per Claim 2, Rotenberg teaches: The microprocessor of claim 1, wherein the branch prediction unit is configured to output the predicted target address in response to a prediction that a branch will be taken (Section 2.1, which discloses "They (the BTB banks) serve the role of detecting branches in the instructions currently being fetched and providing their target addresses, in time for the next fetch cycle. A few paragraphs further down, it is stated that this happens "if there is a predicted taken branch").

11. As per Claim 11, Rotenberg teaches: The microprocessor of claim 1, wherein each of the plurality of traces comprises partially-decoded instructions (it is inherent that a trace comprises a partially-decoded instruction, otherwise the necessary control information as show in section 2.2 would not be available).

12. As per Claim 12, Rotenberg teaches: The microprocessor of claim 1, wherein each of the plurality of traces is associated with a tag comprising the address of an earliest instruction, in program order, stored within that trace (Section 2.2, where the tag identifies the starting address of the trace).

13. As per Claim 13, Rotenberg teaches: The microprocessor of claim 1, wherein each of the plurality of traces is associated with a flow control field comprising a label for an instruction to which control will pass for each branch operation comprised in that trace (Section 2.2, the “trace target address” and “trace fall-through address” are both labels which describe where control will flow based on each branch operation, based on the prediction (which is part of another label, “branch flags”)).

14. As per Claim 14, Rotenberg teaches: A computer system, comprising:

a system memory (inherent if the system has an instruction cache); and

a microprocessor coupled to the system memory (also inherent in Rotenbergs invention), comprising:

an instruction cache configured to store instructions (Section 2.1, first paragraph);

a branch prediction unit (Section 2.1, first paragraph);

a trace cache configured to store a plurality of traces of instructions (Section 2.2); and

a prefetch unit coupled to the instruction cache, the branch prediction unit, and the trace cache (Figure 4, and Section 2.2. The Instruction latch appears to fill the role of a prefetch unit);

wherein the prefetch unit is configured to fetch instructions from the instruction cache until the branch prediction unit outputs a predicted target address (Section 2.2, where it is said that "on a trace cache miss, fetching proceeds normally from the instruction cache); and

wherein if the prefetch unit identifies a match for the predicted target address in the trace cache, the prefetch unit is configured to fetch one or more of the plurality of traces from the trace cache (Section 2.2, where it is said that "on a trace cache hit, an entire trace of instructions is fed into the instruction latch, bypassing the instruction cache).

15. As per Claim 15, Rotenberg teaches: The computer system of claim 14, wherein the branch prediction unit is configured to output the predicted target address in response to a prediction that a branch will be taken (Section 2.1, which discloses "They (the BTB banks) serve the role of detecting branches in the instructions currently being fetched and providing their target addresses, in time for the next fetch cycle. A few paragraphs further down, it is stated that this happens "if there is a predicted taken branch").

16. As per Claim 24, Rotenberg teaches: The computer system of claim 14, wherein each of the plurality of traces comprises partially-decoded instructions (it is inherent that a trace comprises a partially-decoded instruction, otherwise the necessary control information as show in section 2.2 would not be available).

17. As per Claim 25, Rotenberg teaches: The computer system of claim 14, wherein each of the plurality of traces is associated with a tag comprising the address of an earliest instruction, in program order, stored within that trace (Section 2.2, where the tag identifies the starting address of the trace).

18. As per Claim 26, Rotenberg teaches: The computer system of claim 14, wherein each of the plurality of traces is associated with a flow control field comprising a label for an instruction to which control will pass for each branch operation comprised in that trace (Section 2.2, the "trace target address" and "trace fall-through address" are both labels which describe where control will flow based on each branch operation, based on the prediction (which is part of another label, "branch flags")).

19. As per Claim 32, Rotenberg teaches: A method, comprising:

fetching instructions from instruction cache (Section 2.1, paragraphs 1-3);

continuing to fetch instructions from the instruction cache until a branch target address is generated (Section 2.2, where it is said that "on a trace cache miss, fetching proceeds normally from the instruction cache);

if a branch target address is generated, searching trace cache for an entry corresponding to the branch target address (Section 2.2, third paragraph).

20. As per Claim 33, Rotenberg teaches: The method of claim 32, further comprising continuing to fetch instructions from instruction cache if no entry is identified in the trace cache corresponding to the branch target address (Section 2.2, where it is said that "on a trace cache miss, fetching proceeds normally from the instruction cache).

21. As per Claim 34, Rotenberg teaches: The method of claim 32, further comprising fetching one or more traces from the trace cache if an entry is identified in the trace cache corresponding to the branch target address (Section 2.2, where it is said that "on a trace cache hit, an entire trace of instructions is fed into the instruction latch, bypassing the instruction cache).

Claim Rejections - 35 USC § 103

22. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

23. Claims 3, 16 are rejected under 35 U.S.C. 103(a) as being unpatentable over Rotenberg, in view of Patterson et al. (herein Patterson).

As per Claim 3, Rotenberg teaches the microprocessor of claim 1, but fails to teach: wherein the branch prediction unit is configured to output the predicted target address in response to detection of a branch misprediction. However, Patterson teaches that in order to reduce the penalty for a misprediction, you can fetch both the taken and not taken instructions, and put them in the BTB, which would then be able to immediately output the correct path on a misprediction without having to fetch it (Pages 276-277). While it would increase the cost of the system, the advantage is a smaller misprediction penalty, which may worth the cost, depending on the needs of the system. Therefore, one of ordinary skill in the art at the time the invention was made would have stored both the taken and not taken branch paths in the BTB in order to reduce the misprediction penalty, and thus increasing performance.

24. As per Claim 16, Rotenberg teaches the computer system of claim 14, but fails to teach: wherein the branch prediction unit is configured to output the predicted target address in response to detection of a branch misprediction. However, Patterson teaches that in order to reduce the penalty for a misprediction, you can fetch both the taken and not taken instructions, and put them in the BTB, which would then be able to immediately output the correct path on a misprediction without having to fetch it (Pages 276-277). While it would increase the cost of the system, the advantage is a smaller misprediction penalty, which may worth the cost, depending on the needs of the system. Therefore, one of ordinary skill in the art at the time the invention was made would have stored both the taken and not taken branch paths in the BTB in order to reduce the

misprediction penalty, and thus increasing performance.

25. Claims 4, 10, 17, 23 are rejected under 35 U.S.C. 103(a) as being unpatentable over Rotenberg, in view of Braught.

26. As per Claim 4, Rotenberg teaches: The microprocessor of claim 1, further comprising a trace generator (it is necessary for Rotenbergs invention to have a trace generator in order to create traces), but fails to teach: wherein the trace generator is configured to begin a trace with an instruction corresponding to a label boundary. Rotenberg teaches starting a trace on a trace cache miss, which then causes the line-fill buffer to begin filling (Section 2.2, fourth paragraph). The trace cache is only checked for a hit in the case of a branch, as the branch predictions determine if it is a hit or not. Therefore, Rotenbergs invention generates a trace when a branch is encountered, but not necessarily on a label boundary. However, Braught teaches that in Assembly Language, most branches operate on labels, which the machine can only interpret when converted to an address (Page 3). In Braughts example on page 3, all branches are labels, making every branch a label boundary. When combined with Rotenbergs invention, this would mean that a trace would be generated with an instruction corresponding to a label boundary, as all the branches would go to labels, and even an address may be interpreted as a label. Given this knowledge, it would have been obvious to one of ordinary skill in the art that Rotenbergs invention begins traces when it

encounters an instruction with a label boundary, as it only begins on branch instructions, which branch to labels.

27. As per Claim 10, Rotenberg teaches: The microprocessor of claim 4, wherein the trace generator is configured to generate traces in response to instructions being decoded (Section 2.2, the trace can not be completed (written into the cache) until the trace target addresses are calculated, which requires the instructions to be decoded).

28. As per Claim 17, Rotenberg teaches: The computer system of claim 14, further comprising a trace generator (it is necessary for Rotenbergs invention to have a trace generator in order to create traces), but fails to teach: wherein the trace generator is configured to begin a trace with an instruction corresponding to a label boundary. Rotenberg teaches starting a trace on a trace cache miss, which then causes the line-fill buffer to begin filling (Section 2.2, fourth paragraph). The trace cache is only checked for a hit in the case of a branch, as the branch predictions determine if it is a hit or not. Therefore, Rotenbergs invention generates a trace when a branch is encountered, but not necessarily on a label boundary. However, Braught teaches that in Assembly Language, most branches operate on labels, which the machine can only interpret when converted to an address (Page 3). In Braughts example on page 3, all branches are labels, making every branch a label boundary. When combined with Rotenbergs invention, this would mean that a trace would be generated with an instruction corresponding to a label boundary, as all the branches would go to labels, and even an

address may be interpreted as a label. Given this knowledge, it would have been obvious to one of ordinary skill in the art that Rotenbergs invention begins traces when it encounters an instruction with a label boundary, as it only begins on branch instructions, which branch to labels.

29. As per Claim 23, Rotenberg teaches: The computer system of claim 17, wherein the trace generator is configured to generate traces in response to instructions being decoded (Section 2.2, the trace can not be completed (written into the cache) until the trace target addresses are calculated, which requires the instructions to be decoded).

30. Claims 5-8 and 18-21 are rejected under 35 U.S.C. 103(a) as being unpatentable over Rotenberg and Braught, further in view of Lange et al. (USPN 3,896,419, herein Lange).

31. As per Claim 5, Rotenberg and Braught teach the microprocessor of claim 4, but fail to teach: wherein the trace generator is configured to check the trace cache for a duplicate copy of the trace that the trace generator is constructing. However, Lange teaches a system in which a cache is checked for a match with memory, while the memory is being read at the same time. The advantage of Lange's method is no delay in the overall data fetch cycle if there is no match in the cache, and the memory access can be cancelled before it incurs any system slowdown as well (Column 2, Lines 13-31, and Column 5, Lines 5-10). Given this advantage, it would have been obvious to one of

ordinary skill in the art at the time the invention was made to check the trace cache for a duplicate copy of the trace at the same time the trace was being constructed, in order to minimize or eliminate delay in generating the trace in the case that the trace is not in the cache, and aborting the unnecessary trace if it is found in the cache.

32. As per Claim 6, Lange teaches: The microprocessor of claim 5, wherein if the trace generator identifies a duplicate copy of the trace, the trace generator is configured to discard the trace under construction (Column 5, Lines 5-10).

33. As per Claim 7, Rotenberg teaches: The microprocessor of claim 5, wherein if the trace generator identifies an entry corresponding to a duplicate copy of the trace, the trace generator is configured to check the trace cache for an entry corresponding to a next trace to be generated (Section 2.2. The trade cache is checked every time there is a potential new trace, so when one trace is found and discarded, the next potential new trace will cause the trace cache to be checked again).

34. As per Claim 8, Lange teaches: The microprocessor of claim 7, wherein if the trace generator identifies a trace entry corresponding to the next trace to be generated, the trace generator is configured to discard the trace under construction (Column 5, Lines 5-10).

35. As per Claim 18, Rotenberg and Braught teach the computer system of claim 17, but fail to teach: wherein the trace generator is configured to check the trace cache for a duplicate copy of the trace that the trace generator is constructing. However, Lange teaches a system in which a cache is checked for a match with memory, while the memory is being read at the same time. The advantage of Lange's method is no delay in the overall data fetch cycle if there is no match in the cache, and the memory access can be cancelled before it incurs any system slowdown as well (Column 2, Lines 13-31, and Column 5, Lines 5-10). Given this advantage, it would have been obvious to one of ordinary skill in the art at the time the invention was made to check the trace cache for a duplicate copy of the trace at the same time the trace was being constructed, in order to minimize or eliminate delay in generating the trace in the case that the trace is not in the cache, and aborting the unnecessary trace if it is found in the cache.

36. As per Claim 19, Lange teaches: The computer system of claim 18, wherein if the trace generator identifies a duplicate copy of the trace, the trace generator is configured to discard the trace under construction (Column 5, Lines 5-10).

37. As per Claim 20, Rotenberg teaches: The computer system of claim 18, wherein if the trace generator identifies an entry corresponding to a duplicate copy of the trace, the trace generator is configured to check the trace cache for an entry corresponding to a next trace to be generated (Section 2.2. The trade cache is checked every time there is a potential new trace, so when one trace is found and discarded, the next potential

new trace will cause the trace cache to be checked again).

38. As per Claim 21, Lange teaches: The computer system of claim 20, wherein if the trace generator identifies a trace entry corresponding to the next trace to be generated, the trace generator is configured to discard the trace under construction (Column 5, Lines 5-10).

39. Claims 9 and 22 are rejected under 35 U.S.C. 103(a) as being unpatentable over Rotenberg, in view of Akkary et al. (USPN 6,247,121, herein Akkary).

40. As per Claim 9, Rotenberg teaches the microprocessor of claim 4, but fails to teach: wherein the trace generator is configured to generate traces in response to instructions being retired. While Rotenberg teaches that the instructions need to be decoded (Section 2.2) before the trace can be generated, it is not taught that they need to be retired beforehand. Akkary teaches that instructions are not put into the trace buffers until they have been retired, to ensure that they executed correctly (Column 3, Lines 40-44). Rotenberg discusses in Section 2.3 that some traces are committed but never used, thus evicting a useful trace. By ensuring that the trace is correct, the odds that it will be useful is increased, as an incorrect trace should probably not need to be used. Therefore, one of ordinary skill in the art at the time the invention was made would have waiting until an instruction was retired before considering adding it to the trace, in order to ensure accuracy of the trace.

41. As per Claim 22, Rotenberg teaches the computer system of claim 17, but fails to teach: wherein the trace generator is configured to generate traces in response to instructions being retired. While Rotenberg teaches that the instructions need to be decoded (Section 2.2) before the trace can be generated, it is not taught that they need to be retired beforehand. Akkary teaches that instructions are not put into the trace buffers until they have been retired, to ensure that they executed correctly (Column 3, Lines 40-44). Rotenberg discusses in Section 2.3 that some traces are committed but never used, thus evicting a useful trace. By ensuring that the trace is correct, the odds that it will be useful is increased, as an incorrect trace should probably not need to be used. Therefore, one of ordinary skill in the art at the time the invention was made would have waiting until an instruction was retired before considering adding it to the trace, in order to ensure accuracy of the trace.

42. Claims 27-31 and 35 are rejected under 35 U.S.C. 103(a) as being unpatentable over Rotenberg, Braught, in view of Akkary.

43. As per Claim 27, Rotenberg teaches: A method, comprising:

starting construction of a new trace if the received instruction is associated with a branch label (Section 2.2. As shown in the rejections for Claims 4 and 17, Rotenberg starts traces on branches, which branch to labels (or addresses, as shown by Braught), thus obviating that all traces are associated with some branch label), but fails to teach:

receiving a retired instruction;

if the previous trace under construction duplicates a trace in trace cache, delaying construction of the new trace until the received instruction corresponds to a branch label.

While Rotenberg teaches that the instructions need to be decoded (Section 2.2) before the trace can be generated, it is not taught that they need to be retired beforehand. Akkary teaches that instructions are not put into the trace buffers until they have been retired, to ensure that they executed correctly (Column 3, Lines 40-44). Rotenberg discusses in Section 2.3 that some traces are committed but never used, thus evicting a useful trace. By ensuring that the trace is correct, the odds that it will be useful is increased, as an incorrect trace should probably not need to be used. Therefore, one of ordinary skill in the art at the time the invention was made would have waiting until an instruction was retired before considering adding it to the trace, in order to ensure accuracy of the trace.

Lange teaches a system in which a cache is checked for a match with memory, while the memory is being read at the same time. The advantage of Lange's method is no delay in the overall data fetch cycle if there is no match in the cache, and the memory access can be cancelled before it incurs any system slowdown as well (Column 2, Lines 13-31, and Column 5, Lines 5-10). Given this advantage, it would have been obvious to one of ordinary skill in the art at the time the invention was made to check the trace cache for a duplicate copy of the trace at the same time the trace was being constructed, in order to minimize or eliminate delay in generating the trace in the

case that the trace is not in the cache, and aborting the unnecessary trace if it is found in the cache. Therefore, when a trace is found to be in the cache, and the trace generation is aborted, it is obvious that a new trace would not be built until a new branch (which would have to be associated with a branch label) was retired.

44. As per claim 28, Rotenberg teaches: The method of claim 27, further comprising continuing construction of an incomplete trace already in process (Section 2.2).

45. As per Claim 29, Lange teaches: The method of claim 27, further comprising searching trace cache for duplicate entries (Column 5, Lines 5-10).

46. As per Claim 30, Rotenberg teaches: The method of claim 29, further comprising creating a new entry in trace cache if no duplicate entry is identified (Section 2.2).

47. As per Claim 31, Lange teaches: The method of claim 29, further comprising discarding a trace if a duplicate entry is identified (Column 5, Lines 5-10).

48. As per Claim 35, Rotenberg teaches: A microprocessor comprising:

means for starting a new trace if the received operation is a first operation at a branch label (Section 2.2. As shown in the rejections for Claims 4 and 17, Rotenberg starts traces on branches, which branch to labels (or addresses, as shown by Braught), thus obviating that all traces are associated with some branch label), but fails to teach:

means for receiving a retired operation;

means for delaying starting a new trace if the previous trace under construction duplicates a trace in trace cache, until the received operation corresponds to a branch label.

While Rotenberg teaches that the instructions need to be decoded (Section 2.2) before the trace can be generated, it is not taught that they need to be retired beforehand. Akkary teaches that instructions are not put into the trace buffers until they have been retired, to ensure that they executed correctly (Column 3, Lines 40-44). Rotenberg discusses in Section 2.3 that some traces are committed but never used, thus evicting a useful trace. By ensuring that the trace is correct, the odds that it will be useful is increased, as an incorrect trace should probably not need to be used. Therefore, one of ordinary skill in the art at the time the invention was made would have waiting until an instruction was retired before considering adding it to the trace, in order to ensure accuracy of the trace.

Lange teaches a system in which a cache is checked for a match with memory, while the memory is being read at the same time. The advantage of Lange's method is no delay in the overall data fetch cycle if there is no match in the cache, and the memory access can be cancelled before it incurs any system slowdown as well (Column 2, Lines 13-31, and Column 5, Lines 5-10). Given this advantage, it would have been obvious to one of ordinary skill in the art at the time the invention was made to check the trace cache for a duplicate copy of the trace at the same time the trace was being constructed, in order to minimize or eliminate delay in generating the trace in the

case that the trace is not in the cache, and aborting the unnecessary trace if it is found in the cache. Therefore, when a trace is found to be in the cache, and the trace generation is aborted, it is obvious that a new trace would not be built until a new branch (which would have to be associated with a branch label) was retired.

Conclusion

49. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure as follows. Applicant is reminded that in amending in response to a rejection of claims, the patentable novelty must be clearly shown in view of the state of the art disclosed by the references cited and the objections made. Applicant must also show how the amendments avoid such references and objections. See 37 CFR § 1.111(c).

50. Pickett et al. (United States Patent Application Publication 2004/0143721 A1) teaches a trace cache holding partially decoded instructions, which are added to a trace on retirement.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Robert E. Fennema whose telephone number is (571) 272-2748. The examiner can normally be reached on Monday-Friday, 8:00-5:30.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on (571) 272-4162. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Robert E Fennema
Examiner
Art Unit 2183

RF

Eddie C

EDDIE CHAN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100